IN THE CLAIMS:

1. (Previously Presented) A method for operating a data storage system, comprising:

creating a writable virtual disk (vdisk) at a selected time, the writable vdisk referencing changes in data stored in the data storage system after the writable vdisk was created;

maintaining a backing store, the backing store referencing data stored in the data storage system which has not been changed since the writable vdisk was created;

loading blocks of the writable vdisk into a memory, the loaded blocks including a writable vdisk indirect block having a plurality of fields, each field storing a valid pointer to a data block or an invalid pointer representing a particular hole of a plurality of holes, where each hole instructs the data storage system to examine a corresponding virtual block number pointer in the backing store;

loading blocks of the backing store into the memory, the loaded blocks including a backing store indirect block having a plurality of fields, each backing store indirect block field corresponding to a field of the writable vdisk indirect block, one or more backing store indirect block fields having a pointer to a data block;

searching each field of the writable vdisk indirect block for a hole; and

replacing each field having a hole in the writable vdisk indirect block with a new pointer to the data block referenced by the corresponding backing store indirect block field to update the writable vdisk to reference both the data which is unchanged since the writable vdisk was created and the data which has been changed since the writable vdisk was created.

2. (Previously Presented) The method of claim 1, further comprising:

dirtying the data block pointed to by the backing store indirect block to enable write allocation of the dirty data block without altering a data content of the data block.

3. (Previously Presented) The method of claim 1, further comprising:

choosing a new pointer for a newly allocated data block containing an unaltered data content;

setting bits in block allocation structures for the newly allocated data block; and

placing the new pointer to the newly allocated data block into the field of the writable vdisk indirect block to replace the hole.

4. (Previously Presented) The method of claim 3 further comprising:

freeing the dirty data block; and

writing the newly allocated data block to disk.

5. (Previously Presented) The method of claim 4 further comprising:

releasing an association of the writable vdisk to the backing store to thereby separate the writable vdisk data blocks from the backing store data blocks.

6. (Original) The method of claim 1 wherein the pointers contained in the writable vdisk indirect block fields and the backing store indirect block fields comprise logical volume block numbers (VBNs).

7. (Original) The method of claim 1 wherein the invalid pointers contained in the writable vdisk indirect block fields comprise a zero logical volume block number (VBN).

8. (Original) The method of claim 1 wherein the plurality of fields in the writable vdisk indirect block are a writable vdisk level 1 buffer and the plurality of fields in the backing store indirect block are a backing store level 1 buffer.

9. (Previously Presented) An apparatus for operating a computer database, comprising:

a writable virtual disk (vdisk) created at a selected time, the writable vdisk referencing changes in data stored in a data storage system after the writable vdisk was created;

a backing store, the backing store referencing data stored in the data storage system which has not been changed since the writable vdisk was created;

a backdoor message handler adapted to load blocks of the writable vdisk and backing store into a memory of the storage system;

a writable vdisk indirect block in the memory having a plurality of fields, each field storing a valid pointer to a data block or an invalid pointer representing a particular hole of a plurality of holes, where each hole instructs the data storage system to examine a corresponding virtual block number pointer in the backing store;

a backing store indirect block in the memory having a plurality of fields, each backing store indirect block field corresponding to a field of the writable vdisk indirect block, each backing store indirect block field having a pointer to a data block;

a special loading function for searching each field of the writable vdisk indirect block for one or more fields representing a hole; and

a write allocator for replacing each field representing a hole in the writable vdisk indirect block with a new pointer to the data referenced by the corresponding backing store indirect block field to update the writable vdisk to reference both the data which is unchanged since the writable vdisk was created and the data which has been changed since the writable vdisk was created.

10. (Previously Presented) The apparatus of claim 9 wherein the write allocator further comprises:

a new pointer for a newly allocated data block containing an unaltered data content, set bits in block allocation structures for the newly allocated data block, and place the new pointer to the newly allocated data block into the field of the writable vdisk indirect block to replace the hole.

11. (Original) The apparatus of claim 10 wherein the write allocator is further adapted to:

free the dirty data block and write the newly allocated data block to disk.

12. (Original) The apparatus of claim 9 wherein the backdoor message handler loads the blocks of the writable vdisk and the blocks of the backing store during periods of reduced processing activity.

13. (Original) The apparatus of claim 9 wherein the pointers contained in the writable vdisk indirect block fields and the backing store indirect block fields comprise logical volume block numbers (VBNs).

14. (Original) The apparatus of claim 9 wherein the invalid pointers contained in the writable vdisk indirect block fields comprise a zero logical volume block number (VBN).

15. (Original) The apparatus of claim 9 wherein the plurality of fields in the writable vdisk indirect block comprises a writable vdisk level 1 buffer and the plurality of fields in the backing store indirect block comprises a backing store level 1 buffer.

16-18. (Cancelled).

19. (Previously Presented) A data storage system apparatus, comprising:

means for creating a writable virtual disk (vdisk) at a selected time, the writable vdisk referencing changes in data stored in the data storage system after the writable vdisk was created;

means for maintaining a backing store, the backing store referencing data stored in the data storage system which has not been changed since the writable vdisk was created;

means for loading blocks of the writable vdisk into a memory, the loaded blocks including a writable vdisk indirect block having a plurality of fields, each field storing a valid pointer to a data block or an invalid pointer representing a particular hole of a plurality of holes, where each hole instructs the data storage system to examine a corresponding virtual block number pointer in the backing store;

means for loading blocks of the backing store into the memory, the loaded blocks including a backing store indirect block having a plurality of fields, each backing store indirect block field corresponding to a field of the writable vdisk indirect block, one or more backing store indirect block fields having a pointer to a data block;

means for searching each field of the writable vdisk indirect block for a hole; and

means for replacing each field having a hole in the writable vdisk indirect block with a new pointer to the data block referenced by the corresponding backing store indirect block field to update the writable vdisk to reference both the data which is unchanged since the writable vdisk was created and the data which has been changed since the writable vdisk was created .

20. (Previously Presented) A computer readable medium containing executable program instructions executed by a processor, comprising:

program instructions that create a writable virtual disk (vdisk) at a selected time, the writable vdisk referencing changes in data stored in a data storage system after the writable vdisk was created;

program instructions that maintain a backing store, the backing store referencing data stored in the data storage system which has not been changed since the writable vdisk was created;

program instructions that load blocks of the writable vdisk into a memory, the loaded blocks including a writable vdisk indirect block having a plurality of fields, each field storing a valid pointer to a data block or an invalid pointer representing a particular hole of a plurality of holes, where each hole instructs the data storage system to examine a corresponding virtual block number pointer in the backing store;

program instructions that load blocks of the backing store into the memory, the loaded blocks including a backing store indirect block having a plurality of fields, each backing store indirect block field corresponding to a field of the writable vdisk indirect block, one or more backing store indirect block fields having a pointer to a data block;

program instructions that search each field of the writable vdisk indirect block for a hole; and

program instructions that replace each field having a hole in the writable vdisk indirect block with a new pointer to the data block referenced by the corresponding backing store indirect block field to update the writable vdisk to reference both the data which is unchanged since the writable vdisk was created and the data which has been changed since the writable vdisk was created.

21-22. (Cancelled).

23. (Previously Presented) A method for operating a data storage system, comprising:

creating a writable virtual disk (vdisk) at a selected time, the writable vdisk referencing changes in data stored in the data storage system after the writable vdisk was created, the writable vdisk having a plurality of holes where each hole instructs the storage system to examine a corresponding virtual block number pointer in a backing store;

maintaining the backing store, the backing store referencing the data stored in the data storage system which has not been changed since the writable vdisk was created;

searching each field of the writable vdisk for a hole; and

referencing each hole in the writable vdisk to point to the data block referenced by the corresponding backing store indirect block to update the writable vdisk to reference both the data which is unchanged since the writable vdisk was created and the data which has been changed since the writable vdisk was created.

24. (Previously Presented) The method of claim 23, further comprising:

dirtying the data block pointed to by the backing store indirect block to enable write allocation of the dirty data block without altering a data content of the data block.

25. (Previously Presented) The method of claim 23 further comprising:

choosing a new pointer for a newly allocated data block containing an unaltered data content;

setting bits in block allocation structures for the newly allocated data block; and

placing the new pointer to the newly allocated data block into the field of the writable vdisk indirect block to replace the hole.

26. (Previously Presented) The method of claim 25, further comprising:

freeing the dirty data block; and

writing the newly allocated data block to disk.

27. (Previously Presented) The method of claim 26 further comprising:

releasing an association of the writable vdisk to the backing store to thereby separate the writable vdisk data blocks from the backing store data blocks.

28. (Previously Presented) The method of claim 23, further comprising:

including logical volume block numbers (VBNs) in the pointers contained in the writable vdisk indirect block fields and the backing store indirect block fields.

29. (Previously Presented) The method of claim 23, further comprising:

using a zero logical volume block number (VBN) as the invalid pointers contained in the writable vdisk indirect block fields.

30. (Previously Presented) The method of claim 23, further comprising:

using a writable vdisk level 1 buffer for the plurality of fields in the writable vdisk indirect block and using a backing store level 1 buffer for the plurality of fields in the backing store indirect block.

31. (Previously Presented) A data storage system, comprising:

a writable virtual disk (vdisk) created at a selected time, the writable vdisk referencing changes in data stored in the data storage system after the writable vdisk was created, the writable vdisk having a plurality of holes, each hole instructing the storage system to examine a corresponding virtual block number pointer in a backing store;

the backing store referencing data stored in the data storage system which has not been changed since the writable vdisk was created;

a processor to search each field of the writable vdisk for a hole; and

the processor to reference each hole in the writable vdisk to point to the data block referenced by the corresponding backing store indirect block to update the writable vdisk to reference both the data which is unchanged since the writable vdisk was created and the data which has been changed since the writable vdisk was created.

32. (Previously Presented) The system of claim 31, further comprising:

the data block pointed to by the backing store indirect block are dirtied to enable write allocation of the dirty data block without altering a data content of the data block.

33. (Previously Presented) The system of claim 31 further comprising:

a new pointer chosen for a newly allocated data block containing an unaltered data content;

bits are set in a block allocation structures for the newly allocated data block; and

a new pointer to the newly allocated data block placed into a field of the writable vdisk indirect block to replace the hole.

34. (Previously Presented) The system of claim 33, further comprising:

the dirty data block is freed; and

the newly allocated data block is written to disk.

35. (Previously Presented) The system of claim 34 further comprising:

an association of the writable vdisk to the backing store is released to thereby separate the writable vdisk data blocks from the backing store data blocks.

36. (Previously Presented) The system of claim 31, further comprising:

logical volume block numbers (VBNs) included in the pointers contained in the writable vdisk indirect block fields and the backing store indirect block fields.

37. (Previously Presented) The system of claim 31, further comprising:

a zero logical volume block number (VBN) used as the invalid pointers contained in the writable vdisk indirect block fields.

38. (Previously Presented) The system of claim 31, further comprising:

a writable vdisk level 1 buffer used for the plurality of fields in the writable vdisk indirect block and a backing store level 1 buffer used for the plurality of fields in the backing store indirect block.

39. (Previously Presented) A computer readable media containing executable program instructions executed by a processor, comprising:

program instructions that create a writable virtual disk (vdisk) at a selected time, the writable vdisk referencing changes in data stored in a data storage system after the writable vdisk was created, the writable vdisk having a plurality of holes where each hole instructs the storage system to examine a corresponding virtual block number pointer in a backing store;

program instructions that maintain the backing store, the backing store referencing data stored in the data storage system which has not been changed since the writable vdisk was created;

program instructions that search each field of the writable vdisk for a hole; and

program instructions that reference each hole in the writable vdisk to point to the data block referenced by the corresponding backing store indirect block to update the writable vdisk to reference both the data which is unchanged since the writable vdisk was created and the data which has been changed since the writable vdisk was created.

40. (Previously Presented) A method for operating a data storage system, comprising:

creating a writable virtual disk (vdisk) at a selected time, the writable vdisk referencing changes in data stored in the data storage system after the writable vdisk was

created, the writable vdisk having a plurality of holes where each hole instructs the data storage system to examine a corresponding virtual block number pointer in a backing store;

maintaining the backing store, the backing store referencing the data stored in the data storage system which has not been changed since the writable vdisk was created;

searching, by a background task process, each field of the writable vdisk for a hole; and

referencing each hole in the writable vdisk to point to the data block referenced by the corresponding backing store indirect block to update the writable vdisk to reference both the data which is unchanged since the writable vdisk was created and the data which has been changed since the writable vdisk was created.

41. (Previously Presented) A data storage system, comprising:

a writable virtual disk (vdisk) created at a selected time, the writable vdisk referencing changes in data stored in the data storage system after the writable vdisk was created, the writable vdisk having a plurality of holes where each hole instructs the data storage system to examine a corresponding virtual block number pointer in the backing store;

the backing store referencing the data stored in the data storage system which has not been changed since the writable vdisk was created;

a background task processor to search each field of the writable vdisk for a hole; and

the background task processor to reference each hole in the writable vdisk to point to the data block referenced by the corresponding backing store indirect block to update the writable vdisk to reference both the data which is unchanged since the writable vdisk was created and the data which has been changed since the writable vdisk was created.

42. (Previously Presented) A computer readable media containing executable program instructions executed by a processor, comprising:

program instructions that create a writable virtual disk (vdisk) at a selected time, the writable vdisk referencing changes in data stored in a data storage system after the writable vdisk was created, the writable vdisk having a plurality of holes where each hole instructs the data storage system to examine a corresponding virtual block number pointer in a backing store;

program instructions that maintain the backing store, the backing store referencing the data stored in the data storage system which has not been changed since the writable vdisk was created;

program instructions that search, by a background task process, each field of the writable vdisk for a hole; and

program instructions that reference each hole in the writable vdisk to point to the data block referenced by the corresponding backing store indirect block to update the writable vdisk to reference both the data which is unchanged since the writable vdisk was created and the data which has been changed since the writable vdisk was created.